A Strategy for Addressing Ambiguity in Regulatory Requirements

Aaron K. Massey*, Richard L. Rutledge*, Annie I. Antón*, Justin D. Hemmings†, Peter P. Swire†
*School of Interactive Computing, Georgia Institute of Technology, Atlanta, GA, USA
{akmassey, rrutledge, aianton}@gatech.edu
†Scheller College of Business, Georgia Institute of Technology, Atlanta, GA, USA
{Justin.Hemmings, Peter.Swire}@scheller.gatech.edu

***Abstract*—Ambiguities in legal texts can make the difference between regulatory compliance and non-compliance in software systems. Ambiguities are prevalent in laws and regulations. Policy analysts who write laws and regulations and software engineers who build software systems that must comply with laws and regulations approach ambiguity differently. In our prior work, we surfaced differences between the approach taken by policy analysts and technologists in identifying and classifying ambiguities in legal texts. Understanding the rationale behind the identification and classification of legal ambiguities is essential to disambiguating them for requirements engineering. Herein, we discuss a case study in which we seek to understand the rationale used to make determinations about ambiguities in legal texts. Our 48 case study participants identified 373 ambiguities, 99.1% of which were classified using our ambiguity taxonomy. The results of our qualitative analysis suggest participants are consistently able to identify words and phrases they believe to be ambiguous, but are unable to express and agree on a consistent rationale defending their classification. This result supports a strategy for addressing ambiguity in regulatory requirements—software engineers are likely to be successful at identifying components of legal texts that then require supplemental expertise to resolve.**

I. Introduction

> It's now pretty much universally accepted that (a) meaning is inseparable from some act of interpretation and (b) an act of interpretation is always somewhat biased, i.e., informed by the interpreter's particular ideology.
>
> — David Foster Wallace [1]

Software engineers building software systems for deployment in regulated environments must identify, read, understand, interpret, and incorporate relevant regulations into the systems they build. Unfortunately, ambiguities are prevalent in laws and regulations [2]. Requirements engineers have long recognized that ambiguity is an important challenge in specifying and building software systems [3], [4], but many of the approaches developed to mitigate or disambiguate ambiguity in requirements specifications are not appropriate for addressing legal ambiguities. Legal texts cannot be easily re-written; if an ambiguity appears in a current law or regulation, then it must be clarified through interpretation rather than rewording. More importantly, the act of interpretation itself is inherently subjective. Lawyers and engineers bring different, sometimes conflicting, perspectives to the interpretation of legal texts [5].

Some regulatory ambiguities are intentional and accurately describe the policy analyst's intent [6]. When laws are originally written, some ambiguity may be created intentionally to allow the courts to determine later what is appropriate or reasonable. Intentional ambiguities allow laws and regulations to avoid dependence on technologies or practices that could change over time [7], [8]. For example, policy analysts might require that software engineers "make reasonable efforts" to protect access to sensitive information. The details of what constitutes a "reasonable effort" are intentionally open for interpretation because what might have been "reasonable" ten years ago could be "egregious" today.

Previously, we created a legal ambiguity taxonomy to guide both policy analysts and technologists in identifying and classifying ambiguity in regulatory texts that govern software systems [2]. Our earlier work showed that the taxonomy is a useful guide for analyzing ambiguity in regulatory text, yet important questions remain. For example, why did participants not exhibit strong agreement on the type and number of ambiguities present in the legal text [2]? Why were participants willing to indicate that software engineers should be able to implement regulatory requirements in software that were still plagued with unintentional ambiguities [2]? We were unable to address these and other questions because we could not examine the rationale participants used [2].

In this paper, we describe a case study designed to examine the rationale used by individuals identifying and classifying ambiguities from regulatory texts according to our taxonomy. Our 48 case study participants identified 373 ambiguities, 99.1% of which were classified as one of the six defined types in our ambiguity taxonomy. We develop and present ambiguity intensity maps, which are designed to provide a succinct visual summary of the amount and type of ambiguity identified and the impact that ambiguity would have on the requirements engineering process. Our results suggest participants are consistently able to identify ambiguous words and phrases but are unable to express a consistent rationale defending their classification. The lack of a consistent rationale indicates that many of the ambiguities identified would require supplemental expertise to resolve.

The remainder of this paper is organized as follows. Section II introduces related work. Section III summarizes our ambiguity taxonomy. Section IV describes our case study

methodology. Section V details the results of our case study. In Section VI, we discuss the implications of this work. Section VII presents potential threats to the validity of this work. Finally, in Section VIII, we summarize our work and provide directions for future work in this area.

II. Related Work

The majority of software requirements specifications are written in inherently ambiguous natural language [9]. However, software engineers do not yet have a single, comprehensive, accepted definition for ambiguity [7]. Ambiguity has been defined as a statement with more than one interpretation [10]. The *IEEE Recommended Practice for Software Requirements Specifications* states that a requirements specification is unambiguous only when each requirement has a single interpretation [11]. Unfortunately, interpretations are fundamentally tied to the background knowledge of the interpreter, and as a result, they are inevitably biased [1].

If the IEEE definition of an unambiguous statement requires only a single, clear interpretation, then how should we classify statements that have no interpretations? Vague or incomplete statements may not have a valid interpretation. For a requirements engineer, a statement that depends heavily on domain knowledge or terms of art may, at first, appear uninterpretable. Antón et al. explicitly state that incompleteness is a form of engineering ambiguity that must be addressed for policy compliance [12]. Herein, we consider vague or incomplete statements to be ambiguous because they do not have a single, clear interpretation.

Extracting software requirements from regulations is extremely challenging [6], [13], [14]. Simply reading policy documents may be beyond the capability of professional engineers [15]. Yet, identifying ambiguous statements and understanding why those statements are ambiguous are critical skills for requirements engineers specifying regulatory requirements. Even beyond the legal domain, too much unrecognized ambiguity is considered one of the five most important reasons for failure in requirements analysis [9].

In our prior work, we developed an ambiguity taxonomy summarized here in Section III. We examined how 17 case study participants used this taxonomy to identify ambiguity in a legal text. We also examined the types of ambiguities they found and whether they believed those ambiguities should prevent software engineers from implementing software that complies with the legal text. They identified and classified on average 33.47 ambiguities in 104 lines of legal text over the course of 50 minutes [2]. Our analysis revealed that the taxonomy provides adequate coverage (97.5%) of the ambiguities found in the given legal text. The new study we discuss in this paper further validates that our ambiguity taxonomy adequately supports analysis of ambiguities in legal texts. In addition, we explore the rationale participants used while analyzing regulatory ambiguity. To our knowledge, this is the first work that explores the rationale individuals use to identify and classify ambiguous requirements.

Linguists and philosophers often employ a more detailed ambiguity taxonomy than we do herein. Berry et al. identified linguistic ambiguity [7], which they classify according to six broad types, some of which have sub-types. For example, pragmatic ambiguity includes referential ambiguity and deictic ambiguity. Sennet's syntactic classification ambiguity includes the subtypes phrasal, quantifier and operator scope, and pronouns [16]. Similarly, lexical ambiguity could be classified as either homonymy or polysemy [7]. Linguists and philosophers continue to debate the nature of ambiguity and correct usage of natural language [17]. In particular, classifying types of ambiguity is itself often ambiguous [18]. Even seemingly simple grammatical corrections can quickly balloon into fundamental arguments [1]. A discussion of the nuance involved in interpreting or correcting language use is outside the scope of this investigation, but it was important to mention here to dispel the myth that legal ambiguities can be resolved by simply finding the "correct" language.

Researchers differentiate between innocuous ambiguity and nocuous ambiguity [10], [19]. Some statements may be *innocuous* because only one possible interpretation would be reasonable, and these statements are unlikely to lead to misunderstandings [10], [19]. Requirements with statements having more than one reasonable interpretation are *nocuous* and likely to lead to misunderstandings if not clarified [10], [19]. In our prior work, we did not differentiate between innocuous and nocuous ambiguities [2]. Herein, we use participant rationale to determine whether an ambiguity is either (1) innocuous, (2) legally nocuous, (3) technically nocuous, or (4) both legally and technically nocuous–which we refer to as "doubly nocuous."

Some researchers describe approaches to ambiguity that rely on tools and techniques to recognize or eliminate ambiguity in software requirements. For example, Gordon and Breaux use refinements to resolve potential conflicts between regulations from multiple jurisdictions [20]. Matsuoka and Lepage developed a semantic similarity measure using WordNet to automatically identify ambiguous individual terms in software requirements specifications [21]. Herein, our ambiguity taxonomy is the only tool evaluated for use in recognizing ambiguity.

Researchers have used natural language processing to detect and resolve ambiguity in software requirements [22]–[28]. Van Bussel developed a machine learning approach to detecting ambiguity in requirements specifications [27]. Popescu et al. developed a semi-automated process for reducing ambiguity in software requirements using object-oriented modeling [28]. Huertas et al. developed a formal approach for measuring lexical ambiguity in natural language requirements based on automatically accounting for all known definitions of all words in each requirement [25]. Osborne and MacNish use NLP to automatically translate natural language requirements as they are created into a formal representation [22]. Ferrari and Gnesi parse requirements to generate all possible "concept paths," which are used to identify pragmatic ambiguities [26]. Requirements with two or more similar concept paths are considered to be ambiguous [26]. None of these approaches focused on ambiguity in legal texts, however, nor do they quantify or

describe the reasoning policy analysts and technologists use when identifying and classifying ambiguity.

III. Ambiguity Taxonomy

This research relies on an ambiguity taxonomy introduced in our prior work [2] and summarized here for reference. Our ambiguity taxonomy defines six separate types of ambiguity. Table I summarizes these types. The ambiguity types outlined in Table I are not mutually exclusive—it is possible for a single sentence from a legal text to exhibit more than one ambiguity type. Although this ambiguity taxonomy is designed to be broadly applicable, it is not guaranteed to be comprehensive. It is also possible for a sentence to be ambiguous in a way that does not fall into one of these six types. The six defined ambiguity types are based on definitions used in requirements engineering, law, and linguistics. *Lexical* ambiguity refers to a word or phrase with multiple valid meanings. *Syntactic* ambiguity refers to a sequence of words with multiple valid grammatical interpretations regardless of context. *Semantic* ambiguity refers to a sentence with more than one interpretation in its provided context. *Vagueness* refers to a statement that admits borderline cases or relative interpretation. *Incompleteness* is a grammatically correct sentence that provides too little detail to convey a specific or needed meaning. *Referential* ambiguity refers to a grammatically correct sentence with a reference that confuses the reader based on the context provided. Some types of ambiguity require additional analysis or disambiguation before implementation can begin.

Two elements of our taxonomy do not appear in Table I because they are not explicitly defined ambiguity types. We included an *Other* type to represent any words of phrases deemed ambiguous in a way that did not fit into one of the six defined ambiguity types. We also included an *Unambiguous* type to represent paragraphs that have a single, clear interpretation.

IV. Case Study Methodology

We chose to examine a portion of the Health Insurance Portability and Accountability Act (HIPAA)[1] as amended by the HITECH Act.[2] Examination of healthcare regulations is an important part of our prior work [2], [29], [30] and continues to be a motivating domain for regulatory compliance. The number of complaints received by U.S. Department of Health and Human Services (HHS) has increased each of the last five years for which data is available.[3] Penalties for violations can be severe. One data breach resulted in a $4.8 million settlement, and the parties involved also had to pay to implement a corrective action plan.[4] The remainder of this section first describes our participants and materials in Section IV-A and then describes our analysis methods in Section IV-B.

[1]Pub. L. No. 104–191, 110 Stat. 1936 (1996)

[2]Pub. L. No. 111–5, 123 Stat. 115 (2009)

[3]http://www.hhs.gov/ocr/privacy/hipaa/enforcement/data/complaintsyear.html

[4]http://www.hhs.gov/news/press/2014pres/05/20140507b.html

A. Study Participants and Materials

A combination of undergraduate and graduates students enrolled at the Georgia Institute of Technology participated in our case study. They were recruited from multiple sections of two courses entitled 'Legal Aspects of Business' and 'Information Security Strategies and Policies' and offered during the 2015 spring semester by the Scheller College of Business. Fifty students volunteered to participate in one of six study sessions.

Our case study materials consisted of a short tutorial followed by the participant survey. During the tutorial, we described the motivation for this research, explained the ambiguity taxonomy, and defined each ambiguity type using illustrative examples. We then presented a worked example of a legal text[5] to provide participants with an experience as similar as possible to the survey and to allow us to demonstrate each type of annotation that might be required. The tutorial took 20 minutes and was delivered from a script to provide uniform instruction to each session.

After the tutorial, participants began the survey. All participants had 50 minutes to compete the survey. The first question asked the participant to self-identify as one of the following two roles:

1) I am a **technologist**, and I am more interested in creating, building, or engineering software systems than I am in legal compliance or business analysis.
2) I am a **policy analyst**, and I am more interested in regulatory compliance than I am in building technologies.

In prior work [2], we quantitatively surveyed ambiguities identified and classified from 23 paragraphs of legal text from the HITECH Act, 45 CFR Subtitle A, § 170.302. This section specifies the certification criteria for electronic health record systems (EHRs) under Meaningful Use Stage 1. Compliance with this regulation is a required qualification for the HITECH incentives that depend upon the use of a certified EHR. Non-compliance with this regulation could result in both regulatory penalties and loss of marketplace reputation.

Informed by our prior results [2], we selected five paragraphs from § 170.302 for this case study. They were selected because the number and type of ambiguities identified in our prior case study indicated that they would provide clarity for edge cases and insight for discussion points from that work. We provided the complete text of § 170.302 to allow participants to view these five paragraphs in their original contexts. Each paragraph was introduced by a reference sheet that contained the text under examination and any cross-referenced text that was at most one reference away (i.e., If the cross-referenced text itself included a cross-reference, we did not provide this text for disambiguation.). Each of the five paragraphs was followed by five response sheets, for a total of 25 response sheets per participant. Response sheets include the paragraph under examination along with a response block. Each response sheet required that only one identified ambiguity be identified, classified, and discussed. Any participant needing more than

[5]The example legal text was 45 CFR Subtitle A, § 164.312(a).

Table I
CASE STUDY AMBIGUITY TAXONOMY [2]

Ambiguity Type	Definition	Example
Lexical	A word or phrase with multiple valid meanings	Melissa walked to the bank.
Syntactic	A sequence of words with multiple valid grammatical interpretations regardless of context	Quickly read and discuss this tutorial.
Semantic	A sentence with more than one interpretation in its provided context	Fred and Ethel are married.
Vagueness	A statement that admits borderline cases or relative interpretation	Fred is tall.
Incompleteness	A grammatically correct sentence that provides too little detail to convey a specific or needed meaning	Combine flour, eggs, and salt to make fresh pasta.
Referential	A grammatically correct sentence with a reference that confuses the reader based on the context	The boy told his father about the damage. He was very upset.

five response sheets for a given paragraph could obtain them from the survey proctor.

For each response sheet, participants were instructed to underline the words or phrases in the paragraph under examination and then classify it in the response block. The response block included a series of objective and descriptive questions about the identification and classification of each ambiguity. Participants selected the most appropriate ambiguity type from our taxonomy for the classifications. However, in contrast to our prior work, participants were also asked to provide a short, written description of their rationale for their ambiguity type selection. They also indicated whether they thought the identified ambiguity accurately reflected the author's intent.

As in our prior work, we asked participants to agree or disagree with the following statement for each ambiguity:

> Software engineers should be able to build software that complies with this legal text.

We sought to determine whether and how an identified ambiguity affected the implementability of the legal requirement. In our previous study, some participants were willing to accept paragraphs as implementable that they also identified as containing an unintentional ambiguity [2]. This scenario is interesting because it is unintuitive to indicate that an accidental ambiguity would not affect whether a regulatory requirement can be implemented in software. To better understand accidental ambiguities of this kind, we also asked the participants to provide a short, written description of their rationale for their decision to agree or disagree with the statement above.

To mitigate the risk of survey fatigue reducing the detail provided in participant rationale descriptions for paragraphs appearing later in the study, half of the participants received a survey packet with the five legal paragraphs presented in reverse order. The complete text of § 170.302 always appeared prior to the response sheets regardless of the presentation order for the paragraphs under examination.

B. Study Analysis

We now describe our participant inclusion and exclusion criteria, data cleaning process, and employed analysis techniques. Three co-authors of this paper served as subject matter experts[6] and examined each ambiguity our participants identified. We reviewed the collected raw data for errors, omissions, and extraneous marks. Fifty students volunteered to participate in our study and identified 388 ambiguities. One subject did not provide rationales for any identified ambiguities or implementability decisions. Another subject only provided a rationale for one identified ambiguity and found all five legal paragraphs as unimplementable with the repeated rationale "It's impractical to require programmers to have legal knowledge." Although this statement raises questions about the ethical obligations of professional software engineers, it reflects more of the subject's general attitude than their specific rationale for the paragraphs examined in this case study. Responses from both of these subjects were removed from further analysis, leaving us with 48 study participants.

We also examined participants' individual response sheets. Some respondents failed to provide rationales for ambiguity type and implementability for individual paragraphs but responded fully to other paragraphs. In these cases, only the blank response sheets were removed. Also, several participants identified a series of ambiguities for a given paragraph, and then terminated the series with a response sheet marked unambiguous. If any portion of a paragraph is ambiguous, then the entire paragraph is ambiguous. Therefore, we removed these extraneous 'unambiguous' response sheets. Finally, one participant marked a legal paragraph as 'unintentionally unambiguous,' which would indicate that the text was accidentally written to have a single, clear meaning. We removed this response as an outlier. This data cleaning process resulted in 373 identified and classified ambiguities.

[6]Our subject matter experts were an academic software engineering researcher, a professional software engineer with 20 years of experience, and a legal scholar.

In 14 instances, we found inconsistently marked response sheets. If the intention of the participant was clear, we revised the response sheet accordingly. For example, several participants did not underline any words, but did select an ambiguity type and provide supporting rationale using specific phrases from the legal text. We treated these responses as if the phrases had been underlined, and they were included in our analysis.

Finally, we reviewed each participant's rationale for their ambiguity classification and their rationale for whether they believed the paragraph was implementable in software. If two or more of the subject matter experts believed that the participant's ambiguity rationale was not something a typical professional would be able to disambiguate, then that ambiguity was deemed legally nocuous. If two or more of the subject matter experts believed that the participant's implementation rationale was not tacit knowledge for a typical professional, then that ambiguity was deemed technically nocuous. We refer to ambiguities that were both legally and technically nocuous as "doubly nocuous."

To calculate a nocuous score for a paragraph, we first calculated the percentages of its identified ambiguities that were innocuous, legally nocuous (L), technically nocuous (T), and doubly nocuous (D). The nocuous score (N) is then given as:

$$N = \frac{1}{2}L + \frac{1}{2}T + D$$

The intuition behind the score is that innocuous ambiguities do not contribute to the score and doubly nocuous ambiguities are weighted to contribute twice as much as either of the singly nocuous ambiguities. A paragraph with only innocuous ambiguities is scored 0.0 and a paragraph with only doubly nocuous ambiguities is scored 1.0

We used several tools to perform our analysis. We encoded the quantitative portions of the subject response sheets, including the ambiguity type, intentionality, and underlined words, for statistical computations with iPython Notebook.[7] We produced Figures 1 and 2 with the R Project for Statistical Computing[8] and python's matplotlib. The analytical iPython Notebook directly generated the ambiguity intensity maps in LaTeX.

V. Results

We now discuss the results of our case study. Section V-A presents the quantitative results found across all five paragraphs and compares the results to those of our previous case study [2]. Section V-B presents the qualitative results from examining responses to each individual paragraph.

A. Quantitative Results

Figure 1 shows the number of ambiguities identified by all 48 participants for each of the paragraphs examined in the case study. Each type of ambiguity is represented by a different color as shown in the legend. Note that if two participants identified the same word or phrase as ambiguous and classified that ambiguity as the same type, we still treat these as distinct ambiguities rather than counting them as occurrences of the same ambiguity. We do this in part because discernment of occurrences in this study would require direct comparison of participant rationale, which is error-prone. For example, in some cases two or more participants identified the same word or phrase as the same type of ambiguity for subtly different reasons.

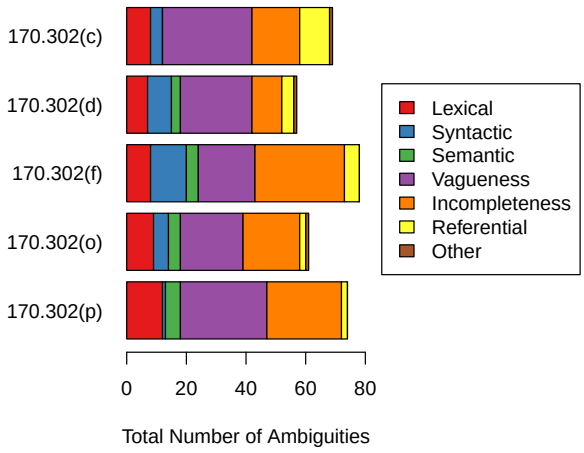

Figure 1. Ambiguities Identified per Case Study Paragraph

The results from Figure 1 are similar to the results found in our previous study for these five paragraphs. The most significant difference between the two studies exists in the numbers of lexical ambiguities and semantic ambiguities identified for both paragraph § 170.302(o) and § 170.302(p). Previously, neither paragraph was found to contain ambiguities of either type, but in this study they were found to contain 21 lexical and 9 semantic. We believe that by asking participants to examine only five paragraphs of legal text and state a rationale defending their classification, they proceeded more carefully and thoughtfully than in the previous study.

Figure 2 displays the ambiguities that participants identified and classified per ambiguity type, and portrays the number of ambiguities classified as intentional or unintentional. Each type (labeled on the x-axis) is represented with two vertical bars. The bar on the left (with hash marks) represents the number of ambiguities identified by technologists, and the one on the right (without hash marks) represents the number identified by policy analysts. Each bar is divided into two parts. The lower part (with a lighter shade) represents the proportion of the total that are unintentional ambiguities, and the upper part (with a darker shade) represents the proportion of intentional ambiguities.

We observed some minor inconsistencies with our previous study. For example, technologists identified almost twice as many incompleteness-type ambiguities as policy analysts in our previous study [2]. In this study, the disparity between incompleteness was not quite as pronounced as it was in the previous study. Vagueness and incompleteness remained the two most-classified types, but in this study participants

[7] http://ipython.org/notebook.html
[8] http://www.r-project.org/

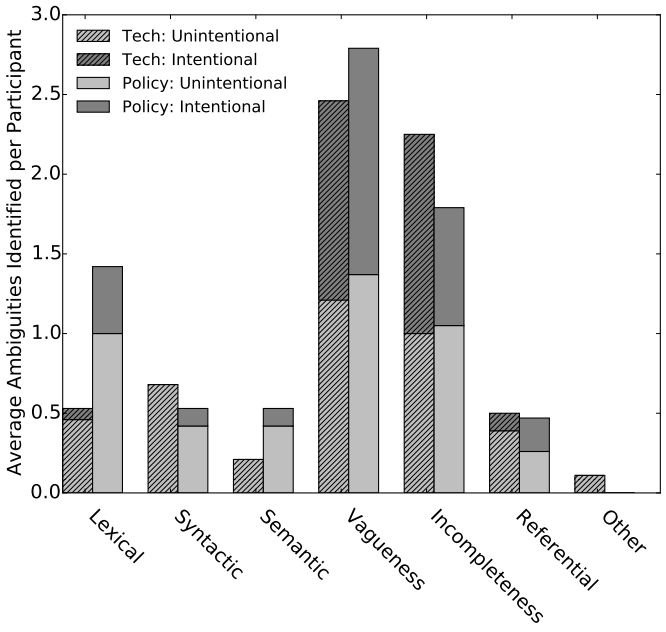

Figure 2. Ambiguities Identified by Technologists and Policy Analysts

identified more vagueness than incompleteness, whereas in the previous study participants identified more incompleteness than vagueness.

The most important difference between the two studies involves the treatment of referential ambiguity. Previously, referential ambiguity was almost as prevalent as incompleteness and decidedly more common than lexical, syntactic, and semantic [2], but in this study it was markedly less prevalent than incompleteness and roughly similar to lexical, syntactic, and semantic. This result was unexpected, particularly since we explicitly chose to include § 170.302(c) in the study because of its high incidence of referential ambiguity in our previous study. Possible explanations for this are discussed along with Figure 3 in Section V-B1.

B. Qualitative Results

To better visualize which sections of the regulatory text were deemed ambiguous by the participants, we created an ambiguity intensity map for each paragraph examined in the case study. Because ambiguity is relative to interpretation, there is no objective approach to determine the "most" ambiguous section in a given text. We can, however, display the relative incidence of ambiguity for words and phrases. Ambiguity intensity maps were inspired by heatmaps or weather maps where the relative intensity of a color indicates the intensity of the underlying data. For our ambiguity intensity maps, a darker shade of red indicates a word or phrase was identified as ambiguous more often than a lighter shade of red. If a word or phrase is not shaded at all, it was not identified as ambiguous by any of the participants. All five figures use the same relative scale, so comparisons of the shading can be made from one figure to another. Beneath the regulatory text in each figure, the ambiguity intensity map includes a simple sparkline-inspired[9] chart to quickly indicate the rates of classification and analysis provided by the participants. The sparkline is divided into four sections. The first section (left side: `L Syn Sem V I R O`) indicates the relative rates of classification for each ambiguity type. The second section (left middle: `Int`) reflects the percentage of ambiguities identified as intentional. The third section (right middle: `U Imp`) reflects the percentages of participants who found a given paragraph to be unambiguous and implementable. The fourth section (right: `Noc`) represents the nocuous score described in Section IV-B. The remainder of this section discusses the results with respect to each paragraph, within the context of each respective ambiguity intensity map.

1) § 170.302(c): In our previous study, § 170.302(c) had the highest proportion of referential ambiguities [2], but it was not clear whether participants classified these ambiguities based on the wording within the paragraph or on the inclusion of cross-references. We chose to include this paragraph in this study to determine whether the cross-references included were responsible for these referential ambiguities.

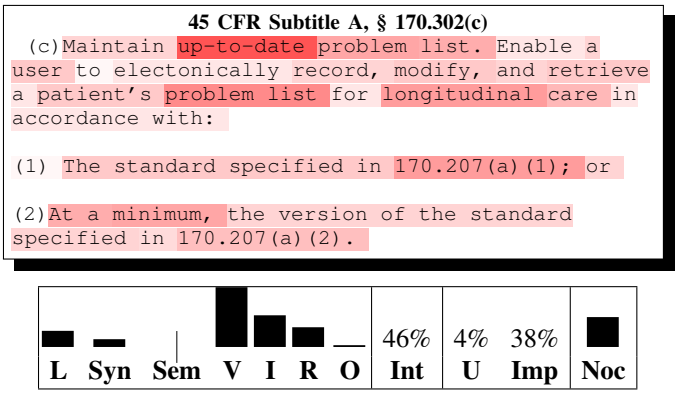

Figure 3. Ambiguity Intensity Map for § 170.302(c)

Figure 3 displays the ambiguity intensity map for § 170.302(c). Certainly, participants classified the cross-references as ambiguous, but, as indicated by the darker shades of red, they identified phrases like "up-to-date" and "problem list" as ambiguous more often than the two cross-references.

By including § 170.302(c) in our case study, we were able to explicitly examine the rationale used when classifying referential ambiguity. In our prior study, participants identified more referential ambiguity in this paragraph than any of the other 22 paragraphs in § 170.302, and cross-references are known to present important compliance challenges for requirements engineers [31]. Unfortunately, the classification rate for referential ambiguity in this paragraph was markedly lower than in the previous study, and we were unable to definitively identify the rationale for identifying referential ambiguity in this paragraph. It is possible that participants were

[9]Sparklines are small graphics designed to emphasize relative rates or tends, rather than convey precise data points. Edward Tufte describes best practices for sparklines here: http://www.edwardtufte.com/bboard/q-and-a-fetch-msg?msg_id=0001OR

experiencing survey fatigue and identified ambiguities for which it would be easier to explain their rationale. It is also possible that as participants identified and classified the phrase "at a minimum," they implicitly included the referential ambiguity as a part of this classification. Those that did indicate a referential ambiguity did so for both of the direct cross-references in the text (e.g., § 170.207(a)(1) and § 170.207(a)(2)). The first cross-referenced text includes another cross-reference, and the second refers to a standard, neither of which were included in this case study. Thus, these ambiguities may simply be an artifact of our need to constrain the case study materials.

Of the 71 ambiguities identified in § 170.302(c), 26 were innocuous, four were legally nocuous, 15 were technically nocuous, and 26 were doubly nocuous. By combining these results as described in Section IV, this paragraph has a nocuous score of 0.50, which was the second highest among the five paragraphs examined in this study and indicates that this paragraph should be prioritized ahead of all but one paragraph for resolution by subject matter experts.

2) § 170.302(d): In our previous study, § 170.302(d) was found to contain a relatively even distribution of ambiguities across all six of our defined types [2]. This even distribution could indicate either a limitation of the taxonomy or a particularly ambiguous, challenging to implement regulatory requirement. We included it in this case study to determine which of these possible explanations best fit with the rationale participants provided as they classified ambiguities in this paragraph.

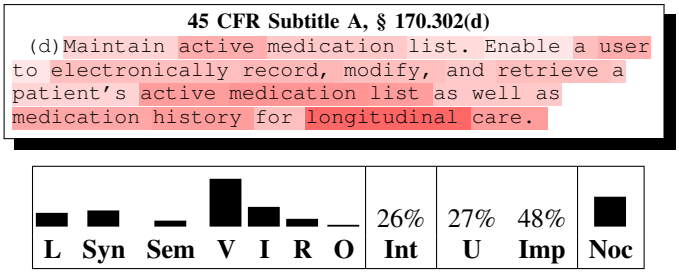

Figure 4. Ambiguity Intensity Map for § 170.302(d)

Figure 4 displays the ambiguity intensity map for § 170.302(d). Participants in this case study noticeably focused on vagueness in this paragraph, which occurred more than twice as often as the next most common ambiguity type. This result conflicts with our prior analysis, which we believe was caused by asking participants to examine only five paragraphs of legal text and to state a rationale defending their classification rather than asking them to examine all 23 paragraphs.

On one hand, 25 of the ambiguities identified in § 170.302(d) were innocuous. The word "longitudinal" was commonly identified as ambiguous either because the definition of the word was unknown or because the time period covered was not precise. Many participants indicated that the ambiguity could be resolved with a definition from a subject matter expert and implementation of the requirement in software would not be inhibited by the need for more precision on this definition.

On the other hand, we also observed 25 doubly nocuous ambiguities. The phrase "Enable a user to electronically record, modify, and retrieve" was identified as ambiguous several times throughout the study.[10] Participants were unclear whether "a user" could refer to a patient, and they expressed concern about whether the user needed to be authorized as indicated in other paragraphs. In addition, participants were not able to determine whether "electronically" applied only to "record" or separately to "record, modify, and retrieve."[11] Disambiguating the interpretation of these phrases is non-trivial, making this ambiguity nocuous for both policy analysts and technologists.

Participants identified a total of 70 ambiguities in this paragraph. In addition to the 25 innocuous and 25 doubly nocuous ambiguities, four were legally nocuous and 16 were technically nocuous. These findings result in a nocuous score of 0.50, which was tied with § 170.302(c) for the second highest score among the five paragraphs examined in this study.

3) § 170.302(f): In our previous study, technologists and policy analysts approached incompleteness differently [2]. Technologists found nearly twice as many incompleteness-type ambiguities as policy analysts [2]. When examining intentional incompleteness, the difference was even more pronounced. Technologists identified an average of 3.2 intentional incompleteness-type ambiguities compared to a 0.1 average for policy analysts [2]. We included § 170.302(f) to confirm this important difference in the ways that technologists and policy analysts perceive ambiguity and to better understand the rationale both groups used when making these classifications.

Figure 2 confirms that the disparity in the number of incompleteness-type ambiguities held true for the five paragraphs included in this study. Participant rationales provide a possible explanation. In several cases, technologists indicated that an ambiguity was incomplete using rationale similar to the rationale a policy analyst would use to classify the same words as vague. This finding is borne out by the data shown in Figure 2, which shows that policy analysts classified more vagueness on average and technologists classified more incompleteness on average.

Figure 5 displays the ambiguity intensity map for § 170.302(f), which indicates that the phrase "at a minimum" stood out as a particularly ambiguous part of this paragraph. Participants indicated that this phrase was incomplete and greatly affected the ability of software engineers to implement compliant software. Participants expressed similar rationale for other ambiguities in this paragraph. For example, the phrase "for patients 2-20 years old" was semantically ambiguous because it did not provide an indication of whether the range was inclusive or exclusive, and this ambiguity greatly affected participant assessment of the implementability of the paragraph.

Participants identified 86 ambiguities in this paragraph, 40 of which were innocuous. Of the remaining ambiguities, 3 were legally nocuous, 24 were technically nocuous, and 19 were doubly nocuous. This paragraph had a nocuous score of 0.38,

[10]This phrase also appears in § 170.302(c) and § 170.302(f).

[11]The application of "electronically" to all verbs in § 170.302(f) is clearly not intended, but the situation is less clear in § 170.302(d).

45 CFR Subtitle A, § 170.302(f)

(f)Record and chart vital signs (1) Vital signs. Enable a user to electronically record, modify, and retrieve a patient's vital signs including, at a minimum, height, weight, and blood pressure.

(2)Calculate body mass index. Automatically calculate and display body mass index (BMI) based on a patient's height and weight.

(3)Plot and display growth charts. Plot and electronically display, upon request, growth charts for patients 2–20 years old.

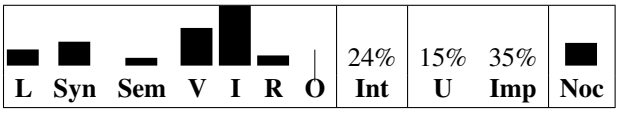

Figure 5. Ambiguity Intensity Map for § 170.302(f)

which was the lowest among the five paragraphs examined in this case study. As a result, the other four paragraphs should take priority in resolution by supplemental expertise.

4) § 170.302(o): One goal of our case study design is to better understand how the intentional use of ambiguity affects the implementability of regulated software. In our previous study, two participants found § 170.302(o) to be both unambiguous and also not implementable [2]. This means that they found that the regulatory text clearly required software engineers to implement something that was impossible to implement. We were particularly interested in the rationale used to come to this conclusion. It might seem unintuitive at first, but the halting problem can be stated unambiguously and not resolved in software. Were these two participants classifying this paragraph as unambiguous and unimplementable because the legal text required something impossible or entirely unrelated to software? We chose to include this paragraph to find out.

45 CFR Subtitle A, § 170.302(o)

(o)Access control. Assign a unique name and/or number for identifying and tracking user identity and establish controls that permit only authorized users to access electronic health information.

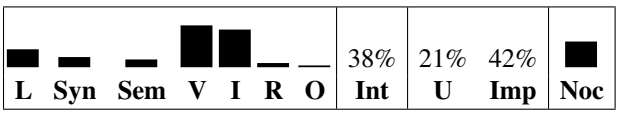

Figure 6. Ambiguity Intensity Map for § 170.302(o)

Figure 6 displays the ambiguity intensity map for § 170.302(o). Participants commonly identified and classified unintended ambiguities while simultaneously stating that software engineers should be able to implement the paragraph in software. Sixteen participants identified at least one ambiguity in this paragraph to be unintentional while simultaneously indicating that the paragraph was implementable. The phrase "unique name and/or number" serves as a common example, and the rationale participants provided indicated that this ambiguity was simply irrelevant to implementation.

The three most ambiguous phrases in this paragraph are "unique name and/or number," "establish controls," and "authorized users." Participant 40 provided an interesting rationale explaining why this last ambiguity is not implementable as written: "Software engineers may think that the system administrator is authorized." Participant 40, a policy analyst, may be seeking to mitigate insider threat issues. Similar rationale raises questions about how to test, debug, or maintain running software systems.

Participants found a total of 71 ambiguities in § 170.302(o). Of these, 29 were innocuous, six were legally nocuous, 17 were technically nocuous, and 19 were doubly nocuous. This paragraph had a nocuous score of 0.43, which was second lowest among the five paragraphs examined in this case study.

5) § 170.302(p): This paragraph contained a high proportion of vagueness in our previous study [2]. We also previously found vagueness to be commonly identified as intentional by both technologists and policy analysts [2]. Because we are interested in how intentional ambiguities affect whether engineers can implement software that complies with laws and regulations, we included this paragraph in the study. In addition, this paragraph contains requirements to allow access to information in "emergency situations," commonly called the "break the glass" scenario. These requirements are of particular interest to EHR vendors.

45 CFR Subtitle A, § 170.302(p)

(p)Emergency access.Permit authorized users (who are authorized for emergency situations) to access electronic health information during an emergency.

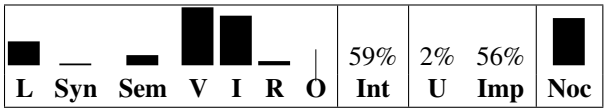

Figure 7. Ambiguity Intensity Map for § 170.302(p)

Figure 7 displays the ambiguity intensity map for § 170.302(p). The "break the glass" scenario was the most commonly identified ambiguity in our case study, as indicated by the dark red sections of Figure 7. Only one participant thought this paragraph was unambiguous, and all 47 of the remaining participants identified the word "emergency" as contributing to an ambiguous phrase in this paragraph. Of these, 38 classified the ambiguity as either vague or incomplete. In total, 26 participants[12] found this paragraph to be implementable in software, indicating that these participants did not consider the "break the glass" ambiguity to affect implementation.

For § 170.302(p), participants found a total of 75 ambiguities, of those seven were innocuous, six were legally nocuous, 12 were technically nocuous, and 50 were doubly nocuous. This paragraph has a nocuous score of 0.79, which was the highest

[12]This number is 27 if we include the participant who found the paragraph to be unambiguous.

of the five paragraphs examined in this case study. It should, therefore, be the first priority for disambiguation, clarification, and resolution by subject matter experts.

VI. Discussion

The results of this work suggest a strategy for addressing ambiguity—software engineers are likely to be successful at identifying components of legal texts that then require supplemental expertise to resolve. Previously, we created an ambiguity taxonomy to guide technologists and policy analysts seeking to identify and classify ambiguity [2]. In this paper, we describe ambiguity intensity maps, which provide a quick visual indicator of which words and phrases are most likely to be misinterpreted and the ambiguity type most likely to be found. We believe these maps can quickly guide an expert to problem areas and improve compliance. Paragraphs in which many individuals agree that certain words or phrases are ambiguous are identifiable by bright red words or phrases. In addition, the sparkline at the bottom of the ambiguity intensity map provides a simple visual breakdown of the types of ambiguity and its implications for technologists seeking to implement the paragraph in software. Ambiguity is dependent upon interpretation, and interpreters all have one bias or another [1]. Describing ambiguity found in a text is challenging because it must be done using relative measures. To our knowledge, our ambiguity intensity map is the first work to visualize ambiguity for the purpose of conveying it to a subject matter expert.

Participant identification and classification of ambiguities in this case study was consistent with our prior work [2] with two notable exceptions. First, participants found proportionally fewer referential ambiguities in this study than in our previous study. This difference was particularly noticeable for § 170.302(c). Second, participants found proportionally more lexical and semantic ambiguities in § 170.302(o) and § 170.302(p). We believe both of these deviations are the result of the differing formats used in each case study. By asking participants to focus on fewer paragraphs of legal text but also to explain their rationale in writing, we may have altered their approach to the problem. It is difficult to say one approach is strictly better than the other, and we expect this to be an area for future research.

Perhaps our most important finding is that participants were unable to express and agree on a consistent rationale defending their identification and classification of ambiguity. This result suggests that many ambiguities found in our case study would require supplemental expertise to resolve. In total, participants identified 373 ambiguities. Of these, only 127 were innocuous and would be readily disambiguated and implemented by technologists and policy analysts. This leaves 23 legally nocuous, 84 technically nocuous, and 139 doubly nocuous ambiguities that would require supplemental subject matter expertise to resolve.

VII. Threats to Validity

Case study research is incomplete without a discussion of concerns that may threaten results validity. *Internal validity* refers to the causal inferences made based on experimental data [32]. Herein, we do not attempt to determine causality for any part of this research. Our goal is to examine and describe the rationale participants use to identify and classify ambiguity in legal texts.

Construct validity refers to the appropriate use of evaluation metrics and measures [32]. We specifically avoided the use of absolute measures of ambiguity to conform with the term as expressed in accepted IEEE standards [11].

Providing participants with only a single section of the HITECH Act and the text of cross-references contained within it is another threat to construct validity. Examining the complete text would have unreasonably increased participant fatigue. However, additional text may have allowed participants to either disambiguate ambiguities identified in our study or discover additional ambiguities resulting from conflicting material.

External validity refers to the ability to generalize the findings to other domains [32]. We mitigated threats to external validity by selecting a section in the HITECH Act that is representative of the style, tone, and wording found throughout the act. We also chose a participant population with as many different backgrounds as possible rather than limiting our research to stakeholders with an engineering background.

Our study population consists of students, rather than practitioners in the healthcare domain. Although our findings are consistent with similar case studies [2], [13], [14], including prior work that demonstrates students are very good indicators of practitioners [13], students may not be accurate representatives of practitioners in the field. Our study also employs a legal text from a single domain: healthcare. Healthcare is a popular domain for regulatory compliance software engineering research, but other domains, like finance, also have extensive regulatory requirements. To address these threats, we plan to adapt what we have learned from this study for a broader, web-based examination of ambiguity identification and classification for multiple legal domains in the future.

Reliability refers to the ability of other researchers to replicate this case study. We carefully detailed our methods and evaluation techniques. In addition, we have made our case study tutorial and survey materials available online for researchers interested in replicating our results.[13]

VIII. Summary and Future Work

Building software systems that are demonstrably compliant with laws and regulations is a critical challenge. It is encouraging for the success of software engineering projects that engineers are effective at spotting the areas of legal text that are ambiguous. Strategies for efficient software engineering can build on that success by making it a specific and early task for engineers to identify areas of potential ambiguity. Where possible, there should be a strategy to elevate areas of legal ambiguity to lawyers, managers, or other subject matter experts to resolve the organization's approach to the ambiguity.

[13] http://www.cc.gatech.edu/~akmassey/documents/ambiguity-rationale-study-materials.pdf

Given the empirical findings in this paper, this division of labor would create an efficient process for identifying, escalating, and resolving legal ambiguities.

The findings here suggest three areas for follow-on research. First, additional empirical work should continue to confirm the finding that software engineers are essentially competent at identifying legal ambiguities but not at resolving them. Second, user studies should be conducted to evaluate the effectiveness of our ambiguity intensity maps at conveying ambiguities identified and classified by software engineers to the qualified subject matter experts capable of resolving them. Third, the proposed strategy of identification, escalation, and resolution of ambiguities should itself be tested for outcomes and compared with alternative approaches. One alternative, which our results suggest will not be as effective, would be for the software engineers to make their best estimate of the meaning of a legal text themselves. Research may suggest additional strategies for organizations to efficiently resolve ambiguities in legal text. The results of this comparative strategy research will be increasingly vital as software for the Internet of Things and control of physical objects becomes more pervasive.

REFERENCES

[1] D. F. Wallace, "Tense present," *Harper's Magazine*, Apr. 2001.

[2] A. K. Massey, R. L. Rutledge, A. I. Antón, and P. P. Swire, "Identifying and classifying ambiguity for regulatory requirements," in *22nd IEEE International Requirements Engineering Conference (RE),* Karlskrona, Sweden, 2014, pp. 83–92.

[3] T. E. Bell and T. A. Thayer, "Software requirements: Are they really a problem?" in *Proceedings of the 2Nd International Conference on Software Engineering*, ser. ICSE '76. Los Alamitos, CA, USA: IEEE Computer Society Press, 1976, pp. 61–68. [Online]. Available: http://dl.acm.org/citation.cfm?id=800253.807650

[4] G. Roman, "A taxonomy of current issues in requirements engineering," *Computer*, vol. 18, no. 4, pp. 14–23, Apr. 1985.

[5] P. Swire and A. Antón, "Engineers and lawyers in privacy protection: Can we all just get along?" *IAPP Privacy Perspectives*, Jan. 2014. [Online]. Available: https://privacyassociation.org/news/a/engineers-and-lawyers-in-privacy-protection-can-we-all-just-get-along/

[6] P. N. Otto and A. I. Antón, "Addressing legal requirements in requirements engineering," *15th IEEE International Requirements Engineering Conference*, pp. 5–14, Oct. 2007.

[7] D. M. Berry, E. Kamsties, and M. M. Krieger, "From contract drafting to software specification: Linguistic sources of ambiguity," School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, Tech. Rep., Nov. 2003.

[8] P. N. Otto, "Reasonableness meets requirements: Regulating security and privacy in software," *Duke Law Journal*, vol. 59, no. 309, pp. 309–342, 2009.

[9] D. M. Berry and E. Kamsties, "Ambiguity in requirements specification," in *Perspectives on Software Requirements*, ser. The Springer International Series in Engineering and Computer Science, P. Leite, J. C. Sampaio, and J. H. Doorn, Eds. Springer US, 2004, vol. 753, pp. 7–44.

[10] F. Chantree, B. Nuseibeh, A. De Roeck, and A. Willis, "Identifying nocuous ambiguities in natural language requirements," in *Requirements Engineering, 14th IEEE International Conference*, 2006, pp. 59–68.

[11] IEEE, *ANSI/IEEE Standard 830-1993: Recommended Practice for Software Requirements Specifications*. Institute of Electrical and Electronics Engineering, New York, NY, 1993.

[12] A. I. Antón, J. B. Earp, and R. A. Carter, "Precluding incongruous behavior by aligning software requirements with security and privacy policies," *Information and Software Technology*, vol. 45, no. 14, pp. 967–977, 2003.

[13] J. C. Maxwell, "Reasoning about legal text evolution for regulatory compliance in software systems," Ph.D. dissertation, North Carolina State University, 2013.

[14] A. Massey, B. Smith, P. Otto, and A. Anton, "Assessing the accuracy of legal implementation readiness decisions," in *19th IEEE International Requirements Engineering Conference (RE)*, Sep. 2011, pp. 207–216.

[15] A. K. Massey, J. Eisenstein, A. I. Antón, and P. Swire, "Automated text mining for requirements analysis of policy documents," in *21st International Conference on Requirements Engineering*, 2013.

[16] A. Sennet, "Ambiguity," in *The Stanford Encyclopedia of Philosophy*, summer 2011 ed., E. N. Zalta, Ed., 2011. [Online]. Available: http://plato.stanford.edu/archives/sum2011/entries/ambiguity/

[17] T. Wasow, A. Perfors, and D. Beaver, "Morphology and the web of grammar: Essays in memory of steven g. lapointe," O. Orgun and P. Sells, Eds. CSLI Publications, 2005.

[18] T. Wasow, "Ambiguity avoidance is overrated," *to appear in a volume edited by Susanne Winkler, http://www.stanford.edu/~wasow/Ambiguity.pdf*, 2014.

[19] H. Yang, A. De Roeck, V. Gervasi, A. Willis, and B. Nuseibeh, "Extending nocuous ambiguity analysis for anaphora in natural language requirements," in *18th IEEE International Requirements Engineering Conference*, 2010, pp. 25–34.

[20] D. Gordon and T. Breaux, "A cross-domain empirical study and legal evaluation of the requirements water marking method," *Requirements Engineering*, vol. 18, no. 2, pp. 147–173, 2013.

[21] J. Matsuoka and Y. Lepage, "Ambiguity spotting using wordnet semantic similarity in support to recommended practice for software requirements specifications," in *2011 7th International Conference on Natural Language Processing andKnowledge Engineering (NLP-KE)*, Nov. 2011, pp. 479–484.

[22] M. Osborne and C. MacNish, "Processing natural language software requirement specifications," in *Requirements Engineering, 1996., Proceedings of the Second International Conference on*, 1996, pp. 229–236.

[23] A. Umber and I. Bajwa, "Minimizing ambiguity in natural language software requirements specification," in *Sixth International Conference on Digital Information Management*, 2011, pp. 102–107.

[24] A. Nigam, N. Arya, B. Nigam, and D. Jain, "Tool for automatic discovery of ambiguity in requirements," *International Journal of Computer Science*, vol. 9, no. 5, Sep. 2012.

[25] C. Huertas, M. Gomez-Ruelas, R. Juarez-Ramirez, and H. Plata, "A formal approach for measuring the lexical ambiguity degree in natural language requirement specification: Polysemes and homonyms focused," in *2011 International Conference on Uncertainty Reasoning and Knowledge Engineering (URKE)*, vol. 1, Aug. 2011, pp. 115–118.

[26] A. Ferrari and S. Gnesi, "Using collective intelligence to detect pragmatic ambiguities," in *Requirements Engineering Conference (RE), 2012 20th IEEE International*, Sep. 2012, pp. 191–200.

[27] D. v. Bussel, "Detecting ambiguity in requirements specifications," Ph.D. dissertation, Tilburg University, Aug. 2009.

[28] D. Popescu, S. Rugaber, N. Medvidovic, and D. Berry, "Reducing ambiguities in requirements specifications via automatically created object-oriented models," in *Innovations for Requirement Analysis. From Stakeholders' Needs to Formal Designs*, ser. Lecture Notes in Computer Science, B. Paech and C. Martell, Eds. Springer Berlin Heidelberg, 2008, vol. 5320, pp. 103–124.

[29] A. K. Massey, P. N. Otto, L. J. Hayward, and A. I. Antón, "Evaluating existing security and privacy requirements for legal compliance," *Requirements Engineering*, 2010.

[30] A. K. Massey, P. N. Otto, and A. I. Antón, "Legal requirements prioritization," in *Proc. of the 2nd Intl. IEEE Workshop on Requirements Engineering and the Law*, 2009.

[31] J. Maxwell, A. Antón, P. Swire, M. Riaz, and C. McCraw, "A legal cross-references taxonomy for reasoning about compliance requirements," *Requirements Engineering*, vol. 17, no. 2, pp. 99–115, 2012. [Online]. Available: http://dx.doi.org/10.1007/s00766-012-0152-5

[32] R. K. Yin, *Case Study Research: Design and Methods*, 3rd ed., ser. Applied Social Research Methods Series, L. Bickman and D. J. Rog, Eds. Sage Publications, 2003, vol. 5.